Fine-grained control of LAMP component versions for Hosting Companies

Xander Lammertink

System & Network Engineering: RP1

Research question

How to migrate customers from LAMP hosting providers to a new type of LAMP hosting where versions of every component can be chosen?

- What is the current situation?
- What should be the situation?
  - Up- or downgrading components?
- How can this be migrated?

What is the current situation?

- Types of hosting
  - Shared hosting
  - Managed hosting
  - Unmanaged hosting
- Resources
- Updates

What software is currently used?

- **LAMP-stack**
  - **L**inux
  - **A**pache
  - **M**ySQL
  - **P**HP

Apache

Microsoft

NGINX

MySQL

php

MariaDB

python

What should be the situation?

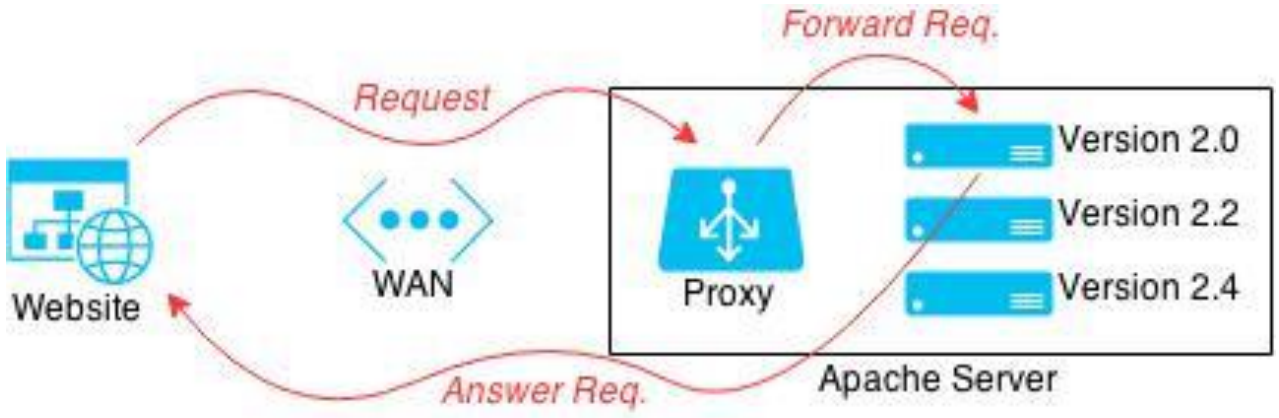
- Control version of components
- Centralized authentication

Linux - Authentication

- Current situation
  - /etc/passwd
  - /etc/shadow
- New situation
  - Centralized
  - Service that handles authentication
  - Kerberos & PAM

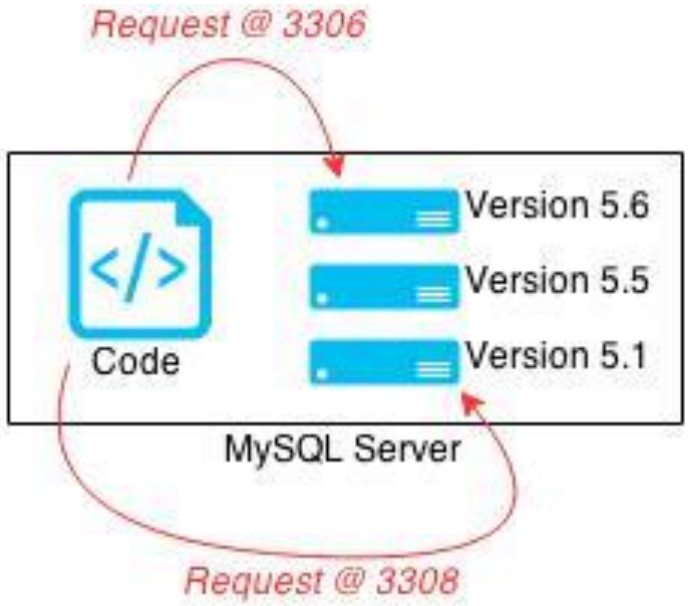
Apache

- Current situation
  - One instance per service
- New situation
  - Multiple instances per server
  - One instance per version
  - One instance as proxy



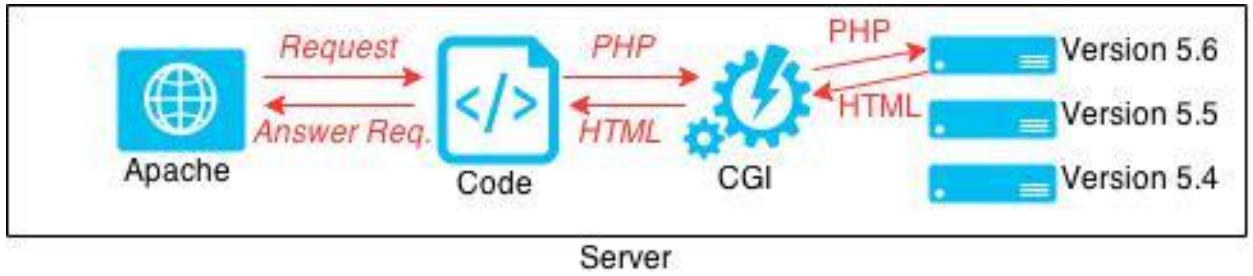
MySQL

- Current situation
  - One instance per server
- New situation
  - Multiple instances per server
  - One instance per version
  - Every instance running on different ports



PHP

- Current situation
  - One instance per server
- New situation
  - Multiple instances per server
  - One instance per version
  - MIME-type to decide version
  - CGI handles communication with PHP
  - .htaccess overwrites global configuration



So tell me…

How to do that?

In a shared hosting environment

Authentication

- Use PAM module
  - pam_krb5_migrate.so
- Copy usernames from /etc/shadow

Apache

- “Setup-instance” script copies current installation
- Setup proxy
  - mod_proxy
- Setup instance

```
<VirtualHost *:80>
    # Setup proxy instance
    ServerName www.example.com
    ProxyPreserveHost On
    ProxyPass / http://127.0.0.1:81/
    ProxyPassReverse / http://127.0.0.1:81/
</VirtualHost>
```

```
<VirtualHost example.com:81>
    # Setup normal instance
    ServerName www.example.com
    DocumentRoot /www/example
</VirtualHost>
```

MySQL

- Install from source
  - Install directory
  - Port number

```
./configure --prefix=[directory] –with-tcp-port=[port]
make
make install
```

- Database is not migrated
  - MySQL Schema Transfer
  - MySQL Dump

PHP

- Apache
  - Create MIME-type
  - Associate MIME-type with CGI
- .htaccess
  - Associate file extension with other MIME-type

```
<Directory “/srv/www”>
   # Create MIME-type
     AddType application/x-httpd-php56 .php56 .php
     AddType application/x-httpd-php55 .php55
   # Associate MIME-type
     Action application/x-httpd-php56 /cgi-bin/php56.cgi
     Action application/x-httpd-php56 /cgi-bin/php55.cgi
</Directory>
```

```
AddType application/x-httpd-php55 .php
```

Conclusion

- All components can run multiple versions
- Usage depends on hosting type

- Authentication
  - Kerberos & PAM
- Apache
  - Proxy for redirection
- MySQL
  - Using different ports
- PHP
  - Via CGI

Suggestions for future work

- Performance
- LAMP using Windows, MariaDB, Pearl, Python, etc.
- Management panels (cPanel, DirectAdmin, etc.)

Questions?!